SLAM3R: Real-Time Dense Scene Reconstruction from Monocular RGB Videos

Yuzheng Liu[1*] Siyan Dong[2*†] Shuzhe Wang[3] Yingda Yin[1]
Yanchao Yang[2†] Qingnan Fan[4] Baoquan Chen[1†]
[1]Peking University [2]The University of Hong Kong [3]Aalto University [4]VIVO

Abstract

*In this paper, we introduce **SLAM3R**, a novel and effective system for real-time, high-quality, dense 3D reconstruction using RGB videos. SLAM3R provides an end-to-end solution by seamlessly integrating local 3D reconstruction and global coordinate registration through feed-forward neural networks. Given an input video, the system first converts it into overlapping clips using a sliding window mechanism. Unlike traditional pose optimization-based methods, SLAM3R directly regresses 3D pointmaps from RGB images in each window and progressively aligns and deforms these local pointmaps to create a globally consistent scene reconstruction - all without explicitly solving any camera parameters. Experiments across datasets consistently show that SLAM3R achieves state-of-the-art reconstruction accuracy and completeness while maintaining real-time performance at 20+ FPS. Code available at: https://github.com/PKU-VCL-3DV/SLAM3R.*

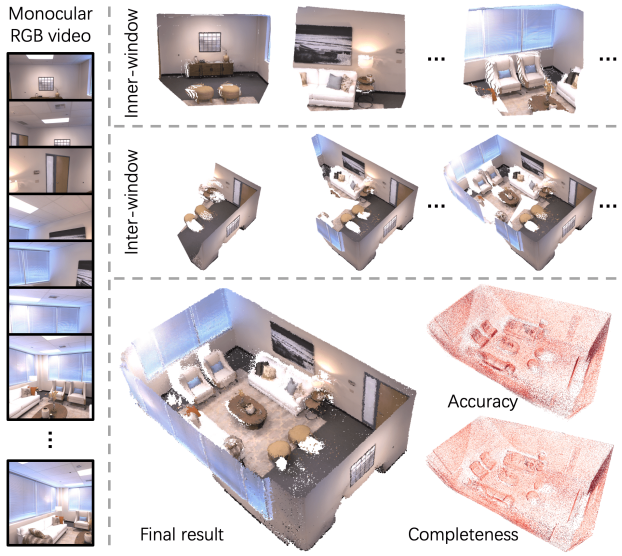

Figure 1. We introduce a novel dense reconstruction system - SLAM3R. It takes a monocular RGB video as input and reconstructs the scene as a dense pointcloud. The video is converted into short clips for local reconstruction (denoted as inner-window), which are then incrementally registered together (inter-window) to create a global scene model. This process runs in real-time, producing a reconstruction that is both accurate and complete.

1. Introduction

Dense 3D reconstruction, a long-standing challenge in computer vision, aims to capture and reconstruct the detailed geometry of real-world scenes. Traditional approaches have largely relied on multi-stage pipelines. These typically begin with sparse Simultaneous Localization and Mapping (SLAM) [7, 16, 25, 37, 38] or Structure-from-Motion (SfM) [31, 33, 47, 53, 66] algorithms to estimate camera parameters, followed by Multi-View Stereo (MVS) [18, 48, 60, 68] techniques to fill in scene details. While these methods offer high-quality reconstructions, they often require offline processing to produce a complete model, which limits their applicability in real-world scenarios.

In the literature, dense SLAM approaches [5, 9, 10, 17, 22, 39, 42, 55, 56, 76, 78] have been developed to address dense scene reconstruction as a complete system. However, these approaches often fall short in terms of reconstruction accuracy or completeness, or they rely heavily on depth sensors. Recently, several monocular SLAM systems [29, 46, 73, 75, 77, 79] have been proposed to tackle dense scene reconstruction from RGB videos. By incorporating advanced scene representations [24, 36, 40, 57, 62], these systems produce accurate and complete scene reconstructions. However, this comes at the cost of reduced running efficiency. For example, NICER-SLAM [79] operates at a speed significantly below 1 FPS. Therefore, current approaches struggle with at least one of three key criteria: reconstruction accuracy, completeness, or efficiency.

While monocular dense SLAM systems encounter the limitations mentioned earlier, recent advances in two-view geometry have shown promising potential. DUSt3R [64] introduces a purely end-to-end approach for learning dense reconstruction. Trained on large-scale datasets, its network

*Joint first authors: liu_yuzheng@stu.pku.edu.cn, siyan3d@hku.hk
†Corresponding authors

is capable of producing high-quality dense reconstructions from paired images in real-time. However, for multiple views, a global optimization step is required to align these image pairs, which significantly hampers its efficiency. A concurrent work, Spann3R [61], extends DUSt3R to multi-view (video) scenarios through an incremental pipeline with spatial memory. While this method accelerates the reconstruction process, it unfortunately results in noticeable accumulated drift and reduced reconstruction quality.

To address these challenges, we present **SLAM3R** (pronounced "slæmər"), a real-time dense 3D reconstruction system using RGB-only videos as input. Unlike traditional SLAM problems, SLAM3R performs implicit camera localization and focuses more on dense scene mapping, where 3R stands for 3D Reconstruction. SLAM3R comprises a two-hierarchy framework. First, it reconstructs local 3D geometry from a sliding window that processes short clips from the input video. Then, it progressively registers these local reconstructions to build a globally consistent 3D scene. Both modules are developed with simple yet effective feed-forward models, enabling end-to-end and efficient scene reconstruction. Specifically, the two modules are the Image-to-Points (I2P) network and the Local-to-World (L2W) network. The I2P module, inspired by DUSt3R, selects a keyframe in a local window as the coordinate system reference. It directly predicts the dense 3D point map supported by the remaining frames within that window. The L2W module incrementally fuses locally reconstructed points into a coherent global coordinate system. Both processes reconstruct the 3D points without explicitly estimating any camera parameters.

Through extensive experiments, we demonstrate that SLAM3R provides high-quality scene reconstructions with minimal drift, outperforming existing dense SLAM systems across various benchmarks. Furthermore, SLAM3R achieves these results at 20+ FPS, bridging the gap between quality and efficiency in RGB-only dense scene reconstruction. Our contributions are summarized below:

- We present a novel real-time end-to-end dense 3D reconstruction system that uses RGB videos to directly predict 3D pointmaps in a unified coordinate system through feed-forward neural networks.
- Through careful design, our Image-to-Points module can process an arbitrary number of images simultaneously, effectively extending DUSt3R to handle multiple views and produce higher-quality predictions.
- The proposed Local-to-World module directly aligns predicted local 3D pointmaps into a unified global coordinate system. This eliminates the need for explicit camera parameter estimation and costly global optimization.
- We evaluate our method on multiple public benchmarks. It achieves state-of-the-art reconstruction quality in terms of both accuracy and completeness at real-time speeds.

2. Related Work

Traditional offline approaches. Dense 3D pointcloud reconstruction is a long-standing problem in computer vision. Classical approaches to this problem first determine camera parameters using Structure from Motion (SfM) [31, 33, 47, 53, 66], followed by dense 3D points triangulation with Multi-View Stereo (MVS) [1, 18, 48, 60, 68]. In recent years, neural implicit [8, 30, 36, 62, 65] and 3D Gaussian [12, 19, 20] representations have been applied to further enhance the quality of dense reconstruction. While these methods deliver high-quality results, they have a significant limitation: the requirement for offline processing to generate the final 3D model, which restricts their applicability in real-time scenarios. In this paper, we focus on online dense reconstruction in the context of Simultaneous Localization and Mapping (SLAM).

Dense SLAM. Early works on SLAM [4, 7, 15, 16, 25, 37, 38] focused on reconstructing the structure of unknown environments while simultaneously localizing camera poses. These approaches prioritize real-time performance but produce only sparse structures of the scene. Dense SLAM approaches [5, 9–11, 17, 22, 27, 39, 42, 55, 56, 76, 78] incorporate detailed scene geometry information to improve pose estimation. DROID-SLAM [56] introduces recurrent iterative updates of camera poses and pixel-wise depth estimates, while TANDEM [27] proposes an online MVS module for depth prediction. These systems enable real-time dense scene reconstruction. However, their focus on camera trajectory accuracy often results in incomplete and noisy 3D reconstruction. Neural implicit and Gaussian representations have also been integrated with dense SLAM systems [9, 21–23, 32, 34, 42, 42, 44, 45, 55, 67, 72, 78]. However, these approaches often rely on additional depth sensors or focus primarily on novel view synthesis rather than producing detailed geometric reconstruction.

More recently, several monocular dense SLAM systems [29, 46, 73, 75, 77, 79] have been developed to produce dense scene geometry reconstruction. A notable limitation of these systems is their slow runtime. Among these systems, GO-SLAM [75] achieves a speed of ∼8 FPS, which still falls short of real-time capability. Furthermore, these methods all share a common strategy: they alternate between solving for camera poses and estimating the scene representation. In contrast, this paper presents a novel approach to dense scene reconstruction that eliminates the need for explicitly solving camera parameters, offering a more efficient and streamlined solution.

End-to-end dense 3D reconstruction. DUSt3R [64] introduces the first purely end-to-end dense 3D reconstruction pipeline without relying on camera parameters. Recently, several works have adopted a similar approach for single-view reconstruction [63], feature matching [28], novel view synthesis [52, 70], and dynamics reconstruction [74]. These

successes demonstrate the effectiveness of end-to-end dense point prediction, inspiring us to develop a dense reconstruction system with a similar methodology.

While DUSt3R operates in real-time for two-view predictions, its extension to multiple views involves exhaustive pairing images and performing an additional global optimization step. This process significantly increases computational time, thereby hindering its real-time performance. MASt3R [28] enhances the matching capability of DUSt3R by adding a match head, achieving more accurate keypoint correspondences for 3D reconstruction [14], but at the cost of increased computational time. More recently, the concurrent work Spann3R [61] extends DUSt3R with spatial memory. It takes a video as input and performs incremental scene reconstruction in a unified coordinate system without requiring global optimization. While this approach significantly improves runtime efficiency, the incremental reconstruction pipeline frame by frame leads to noticeable accumulated drift. Unlike Spann3R, our networks at each hierarchy take multiple frames as input to minimize drift. Additionally, we propose a self-contained retrieval module that, when registering a new frame, this module selects not only its previous few frames but also other similar frames from long-term history for more global scene reference.

3. Method

Problem statement. Given a monocular video consisting of a sequence of RGB image frames $\{I_i \in \mathbb{R}^{H\times W\times 3}\}_{i=1}^{N}$ that captures a static scene, the goal is to reconstruct its dense 3D poincloud $P \in \mathbb{R}^{M\times 3}$, where M is the number of 3D points. Our work focuses on three key objectives: maximizing 3D points recovery for reconstruction completeness, improving the accuracy of each recovered point, and achieving these goals while preserving real-time performance.

System overview. Figure 2 illustrates an overview of the proposed dense reconstruction system. It consists of two main components: an Image-to-Points (I2P) network that recovers local 3D points from video clips, and a Local-to-World (L2W) network that registers local reconstructions into a global scene coordinate system. During the reconstruction of the dense point cloud, the system does not explicitly solve any camera parameters. Instead, it directly predicts 3D point maps in unified coordinate systems.

The system starts by applying a sliding window mechanism of length L to convert the input video into short clips $\{\mathcal{W}_i \in \mathbb{R}^{L\times H\times W\times 3}\}$. The I2P network then processes each window $\mathcal{W}_i$ to recover local 3D pointmaps. Within each window, the system selects a keyframe to define a reference coordinate system for point reconstruction, as detailed in Sec. 3.1. By default, the stride of the sliding window is set to 1, ensuring each input frame in the video is selected at least once as a keyframe. For global scene reconstruction, we initialize the world coordinate system with the first window and use the reconstructed frames (image and local point map produced by the I2P) as input for the L2W model. The L2W model incrementally register these local reconstructions into a unified global 3D coordinate system. To ensure both accuracy and efficiency during this process, the system maintains a limited reservoir of registered frames, called scene frames. Whenever the L2W model registers a new keyframe, we retrieve the best-correlated scene frames as a reference. The details are introduced in Sec. 3.2.

3.1. Inner-Window Local Reconstruction

The Image-to-Points (I2P) model aims to infer dense 3D pointmaps for every pixel of a keyframe in a given video clip. By default, the middle image of a window $\mathcal{W}$ is chosen as the keyframe I_{key} to define the local coordinate system, as it is most likely to have the largest overlap with other frames. The remaining images $\{I_{sup_i}\}_{i=1}^{L-1}$ serve as supporting frames. Note that the 3D pointmaps of supporting frames can also be reconstructed through I2P.

The I2P network draws inspiration from DUSt3R [64], originally designed for stereo 3D reconstruction. We introduce several simple yet effective modifications to extend it for multi-view scenarios. The I2P model uses a multi-branch Vision Transformer (ViT) [13] as its backbone. It consists of a shared encoder E_{img}, two separate decoders D_{key} and D_{sup}, and a point regression head for final prediction. These components are detailed below.

Image encoder. For a given video clip, the image encoder E_{img} encodes each frame I_i to obtain token representations $F_i \in \mathbb{R}^{T\times d}$, where T is the number of tokens and d is the token dimension. The encoder E_{img} comprises m ViT encoder blocks, each containing self-attention and feed-forward layers. The encoding process is denoted as

$$F_i^{(T\times d)} = E_{img}(I_i^{(H\times W\times 3)}),\ i = 1, ..., L.$$

The frames are processed independently and in parallel, with the output divided into two parts: F_{key} for the keyframe and $\{F_{sup_i}\}_{i=1}^{L-1}$ for the supporting frames.

Keyframe decoder. The keyframe decoder D_{key} consists of n ViT decoder blocks, each containing self-attention, cross-attention, and feed-forward layers. Unlike DUSt3R which uses the standard cross-attention, we introduce a novel multi-view cross-attention to combine information from different supporting frames. Given the feature tokens F_{key} and $\{F_{sup_i}\}_{i=1}^{L-1}$, the keyframe decoder D_{key} takes F_{key} as input for self-attention and performs cross-attention between F_{key} and $\{F_{sup_i}\}_{i=1}^{L-1}$. A decoder block is illustrated in Figure 3. For each cross-attention layer, queries are taken from F_{key}, while keys and values are extracted from the supporting tokens F_{sup_i}. These $L-1$ cross-attention layers are independent of each other, allowing for parallel processing. A max-pooling layer is then employed to aggregate features after cross-attention. We obtain decoded

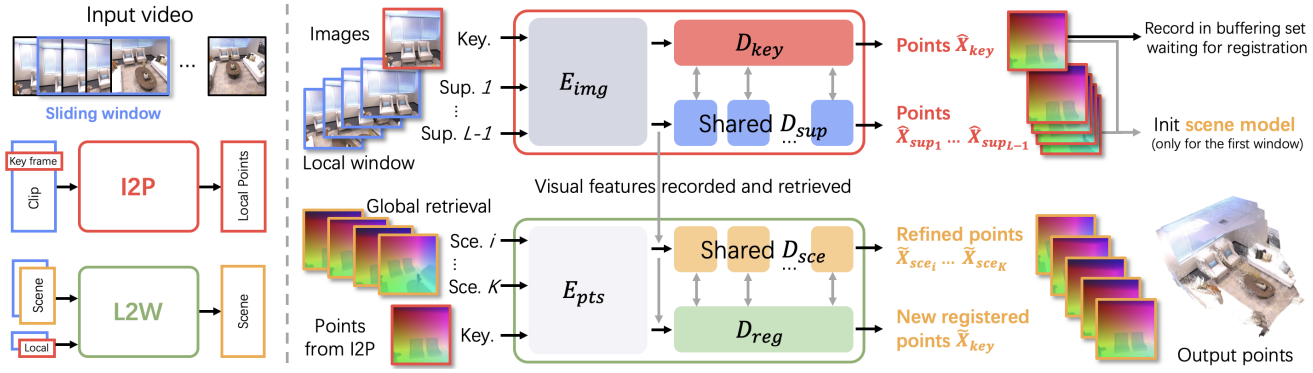

Figure 2. System overview. Given an input monocular RGB video, we apply a sliding window mechanism to convert it into overlapping clips (referred to as windows). Each window is fed into an Image-to-Points (I2P) network to recover 3D points in a local coordinate system. Next, the local points are incrementally fed into a Local-to-World (L2W) network to create a globally consistent scene model. The proposed I2P and L2W networks elegantly share similar architectures. In the I2P step (Sec. 3.1), we select a keyframe as a reference to set up a local coordinate system and use the remaining frames in the window to estimate the 3D geometry captured within it. The points from the first window are used to establish the world coordinate system. We then incrementally fuse the following windows in the L2W step (Sec. 3.2). This process involves retrieving the most relevant already-registered keyframes as a reference, and integrating new keyframes. Through this iterative process, we eventually obtain the full scene reconstruction.

keyframe tokens G_{key} as:

$$G_{key} = D_{key}(F_{key}, F_{sup_1}, ..., F_{sup_{L-1}}).$$

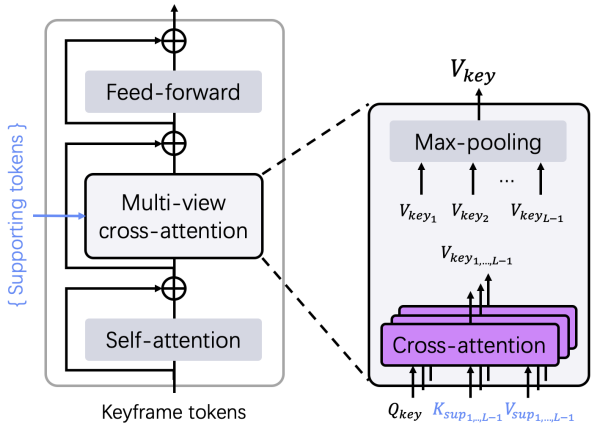

Figure 3. Illustration of a decoder block in the proposed keyframe decoder D_{key}. We present a minimalist modification to integrate information from different supporting images. Our approach traverses each of them, selects its token keys and values, and uses the keyframe queries to interact with them separately across the supporting images. This multi-view information is then aggregated through max-pooling. The registration decoder D_{reg} and scene decoder D_{sce} (described in Sec. 3.2) share the same architecture.

Supporting decoder. The supporting decoder D_{sup} is designed to complement the keyframe decoder. It inherits the decoder architecture used in DUSt3R, consisting of n standard ViT decoder blocks. The cross-attention mechanism is applied only to exchange information with the keyframe. Note that all supporting frames share the same D_{sup}. This process is denoted as

$$G_{sup_i} = D_{sup}(F_{sup_i}, F_{key}),\ i = 1, ..., L-1.$$

Points reconstruction. Similar to DUSt3R, we apply a linear head [64] to regress dense 3D pointmaps in the unified coordinate system from decoded tokens. In addition to the pointmaps, we also predict the confidence maps for all frames to evaluate their reliability. The final predictions are:

$$\hat{X}_i^{(H\times W\times 3)}, \hat{C}_i^{(H\times W\times 1)} = \mathrm{H}(G_i^{(T\times d)}),\ i = 1, ..., L.$$

Training loss. Following DUSt3R, the I2P network is trained end-to-end using ground-truth scene points $\{X_i\}_{i=1}^{L}$. Both the ground truth and predicted point maps are normalized to a canonical scale, determined by the average distance of all valid points within the window to the origin. The confidence-aware training loss is:

$$\mathcal{L}_{I2P} = \sum_{i=1}^{L} M_i \cdot (\hat{C}_i \cdot \mathrm{L1}(\frac{1}{\hat{z}}\hat{X}_i, \frac{1}{z}X_i) - \alpha \log \hat{C}_i),$$

where M_i represents a mask of valid points that have ground-truth values in X_i, z and $\hat{z}$ are the scale factor, $\hat{C}_i$ is the confidence map, the operator $\cdot$ denotes the element-wise matrix multiplication, $\mathrm{L1}(\cdot)$ denotes the point-wise Euclidean distance, and α is a hyper-parameter to control the regularization term. We will detail the process in Sec 4.

3.2. Inter-Window Global Registration

After obtaining the 3D pointmap $\{\hat{X}_{key}\}$ from the I2P network, we use the inter-window Local-to-World (L2W)

model to incrementally register the newly generated pointmap into a global 3D coordinate system. Similar to the I2P network, the L2W model also relies on some frames to serve as a reference for the scene coordinate system. Furthermore, it can leverage multiple registered keyframes as a global reference. These registered keyframes are referred to as scene frames and they are maintained in a buffering set through a sampling mechanism.

The buffering set is designed for scalability in handling long videos. We apply a reservoir strategy [59] that maintains a maximum of B registered frames in the buffering set. When a new keyframe is inferred from I2P and ready for fusion, we retrieve the top-K best-correlated scene frames from the buffering set as its support for global registration.

Scene initialization. The first window is used to initialize the scene model. It's crucial to ensure this initialization is as accurate as possible. To achieve this, we execute the I2P network L times, attempting to traverse and designate each frame within the window as the keyframe. We then select the result with the highest total confidence score for scene model initialization. This process results in a scene pointcloud along with a set of registered frames. All these frames are regarded as scene frames, and they are used to initialize the buffering set.

Reservoir and retrieval. Each scene frame I_{sce_i} is recorded with its latent feature F_{sce_i} and pointmaps $\tilde{X}_{sce_i}$. For efficiency, we apply reservoir sampling to allow storing an unbiased subset of an empirical distribution in a bounded amount of memory. The first B registered frames chosen are directly inserted into the buffering set. For each subsequent frame with $id > B$, the probability of inserting it is B/id. If chosen for insertion, it will randomly replace one of the current scene frames in the buffering set.

Given a new keyframe I_{key} to be registered, we feed its feature F_{key} and the features from the buffering set into a retrieval module,

$$\text{Retrieval}(F_{key}^{(T\times d)}, \{F_{sce_i}^{(T\times d)}\}),$$

to obtain a list of correlation scores, measuring both the visual similarity and baseline suitability between the keyframe and the scene frames in the buffering set. The retrieval module uses the first r decoder blocks from the I2P module as its backbone. A linear projection and an average-pooling layer follow, together producing an image-wise correlation score. We then select the top-K scene frames as a global reference to fuse the current keyframe. As a result, we have K scene frames and one keyframe as the input for the following L2W model.

Points embedding. The 3D pointmaps reconstructed by the I2P model are encoded into the L2W model using a patch embedding method similar to image patchification in the ViT encoder E_{img}. We process the new keyframe and K retrieved scene frames in parallel as:

$$\mathcal{P}_i^{(T\times d)} = E_{pts}(\hat{X}_i^{(H\times W\times 3)}),\ i = 1, ..., K+1.$$

The encoded geometric tokens are combined with their corresponding visual tokens by

$$\mathcal{F}_i^{(T\times d)} = F_i^{(T\times d)} + \mathcal{P}_i^{(T\times d)},\ i = 1, ..., K+1.$$

This resulting a token set $\{\mathcal{F}_{key}, \{\mathcal{F}_{sce_i}\}_{i=1}^K\}$ contains joint features of image patch appearance and 3D geometry for the keyframe and retrieved scene frames. In the following decoders, $\{\mathcal{P}\}$ are further accumulated to $\{\mathcal{F}\}$ between adjacent blocks to enhance the geometric representation.

Registration decoder. The registration decoder D_{reg} takes feature tokens $\{\mathcal{F}_{key}, \{\mathcal{F}_{sce_i}\}_{i=1}^K\}$ as input and aims to transform the local reconstruction of the keyframe to the scene coordinate system. It takes the same network architecture of the keyframe decoder D_{key}. This decoding process is denoted by

$$\mathcal{G}_{key} = D_{reg}(\mathcal{F}_{key}, \mathcal{F}_{sce_1}, ..., \mathcal{F}_{sce_K}).$$

Scene decoder. The scene decoder D_{sce} takes the token set $\{\mathcal{F}_{key}, \{\mathcal{F}_{sce_i}\}_{i=1}^K\}$ as input to refine the scene geometry without coordinate system changes. It uses the same network architecture as the keyframe decoder D_{key}, allowing us to extend to multi-keyframe co-registration (see supplementary material for details). By default, we register one keyframe each time. Each of the $\mathcal{F}_{sce_i}$ has information exchange only with the $\mathcal{F}_{key}$. This decoding process is denoted by

$$\mathcal{G}_{sce_i} = D_{sce}(\mathcal{F}_{sce_i}, \mathcal{F}_{key}),\ i = 1, ..., K.$$

Points reconstruction and training loss. We apply the same head design as that of the I2P network to predict all the pointmaps $\tilde{X}_i$ in the global scene coordinate system:

$$\tilde{X}_i^{(H\times W\times 3)}, \tilde{C}_i^{(H\times W\times 1)} = \text{H}(\mathcal{G}_i^{(T\times d)}),\ i = 1, ..., K+1.$$

We train the L2W network using a similar loss function as the I2P network. Differently, no normalization is applied to the predicted point map, as the output scale must align with the scene frames in the input. This alignment ensures that the output can be directly integrated into the existing reconstruction. The training loss of the L2W network is:

$$\mathcal{L}_{L2W} = \sum_{i=1}^{L} M_i \cdot (\tilde{C}_i \cdot \text{L1}(\tilde{X}_i, X_i) - \alpha \log \tilde{C}_i).$$

The following section provides a detailed discussion of the training process and its implementation details.

Method	Chess Acc. / Comp.	Fire Acc. / Comp.	Heads Acc. / Comp.	Office Acc. / Comp.	Pumpkin Acc. / Comp.	RedKitchen Acc. / Comp.	Stairs Acc. / Comp.	Average Acc. / Comp.	FPS
DUSt3R [64]	2.26 / 2.13	**1.04** / 1.50	1.66 / **0.98**	4.62 / 4.74	**1.73 / 2.43**	**1.95 / 2.36**	3.37 / **10.75**	**2.19 / 3.24**	<1
MASt3R [28]	**2.08 / 2.12**	1.54 / **1.43**	**1.06** / 1.04	**3.23 / 3.19**	5.68 / 3.07	3.50 / 3.37	**2.36** / 13.16	3.04 / 3.90	≪1
Spann3R [61]	2.23 / 1.68	0.88 / 0.92	**2.67 / 0.98**	5.86 / 3.54	2.25 / **1.85**	2.68 / **1.80**	5.65 / **5.15**	3.42 / 2.41	>50
SLAM3R-NoConf (Ours)	2.12 / **1.21**	0.95 / **0.80**	3.23 / 1.67	2.59 / **2.21**	1.99 / 2.04	2.09 / 1.88	4.54 / 6.38	2.40 / **2.24**	∼25
SLAM3R (Ours)	**1.63** / 1.31	**0.84** / 0.83	2.95 / 1.22	**2.32** / 2.26	**1.81** / 2.05	**1.84** / 1.94	**4.19** / 6.91	**2.13** / 2.34	∼25

Table 1. Reconstruction results on 7 Scenes [51] dataset. The average numbers are computed over all test sequences. The methods are categorized into two groups based on whether their FPS is above or below 1. The best results within each category are shown in bold. We report accuracy and completeness in centimeters. The color gradient shifts from red through yellow to green to show increasing FPS.

Method	Room 0 Acc. / Comp.	Room 1 Acc. / Comp.	Room 2 Acc. / Comp.	Office 0 Acc. / Comp.	Office 1 Acc. / Comp.	Office 2 Acc. / Comp.	Office 3 Acc. / Comp.	Office 4 Acc. / Comp.	Average Acc. / Comp.	FPS
DUSt3R [64]	3.47 / **2.50**	**2.53 / 1.86**	**2.95 / 1.76**	4.92 / 3.51	**3.09 / 2.21**	4.01 / 3.10	**3.27 / 2.25**	3.66 / 2.61	**3.49 / 2.48**	<1
MASt3R [28]	4.01 / 4.10	3.61 / 3.25	3.13 / 2.15	**2.57 / 1.63**	12.85 / 8.13	**3.13 / 1.99**	4.67 / 3.15	3.69 / **2.47**	4.71 / 3.36	≪1
NICER-SLAM [79]*	**2.53** / 3.04	3.93 / 4.10	3.40 / 3.42	5.49 / 6.09	3.45 / 4.42	4.02 / 4.29	3.34 / 4.03	**3.03** / 3.87	3.65 / 4.16	≪1
DROID-SLAM [56]*	12.18 / 8.96	8.35 / 6.07	3.26 / 16.01	**3.01** / 16.19	**2.39** / 16.20	5.66 / 15.56	4.49 / 9.73	4.65 / 9.63	5.50 / 12.29	∼20
DIM-SLAM [29]*	14.19 / 6.24	9.56 / 6.45	8.41 / 12.17	10.16 / 5.95	7.86 / 8.33	16.50 / 8.28	13.01 / 6.77	13.08 / 8.62	11.60 / 7.85	∼3
GO-SLAM [75]	-	-	-	-	-	-	-	-	3.81 / 4.79	∼8
Spann3R [61]	9.75 / 12.94	15.51 / 12.94	7.28 / 8.50	5.46 / 18.75	5.24 / 16.64	9.33 / 11.80	16.00 / 9.03	13.97 / 16.02	10.32 / 13.33	>50
SLAM3R-NoConf (Ours)	3.37 / **2.40**	3.22 / **2.33**	3.15 / **2.00**	4.43 / **2.59**	3.18 / **2.34**	3.95 / **2.78**	4.20 / **3.15**	4.57 / 3.38	3.76 / **2.62**	∼24
SLAM3R (Ours)	**3.19 / 2.40**	**3.12** / 2.34	**2.72 / 2.00**	4.28 / 2.60	3.17 / **2.34**	**3.84 / 2.78**	**3.90** / 3.16	**4.32 / 3.36**	**3.57 / 2.62**	∼24

Table 2. Reconstruction results on Replica [54] dataset. * denotes the results reported in NICER-SLAM.

	DUSt3R [64]	MASt3R [28]	NICER-SLAM [79]*	DROID-SLAM [56]*	DIM-SLAM [29]	GO-SLAM [75]	Spann3R [61]	**SLAM3R-NoConf (Ours)**	**SLAM3R (Ours)**
7 Scenes	8.02	**6.28**	8.55	**5.66**	-	-	11.70	8.44	8.41
Replica	4.76	**1.67**	1.88	**0.33**	0.46	0.39	32.79	6.61	6.61

Table 3. Camera pose results evaluated by ATE-RMSE (cm) on 7 Scenes [51] and Replica [54] datasets.

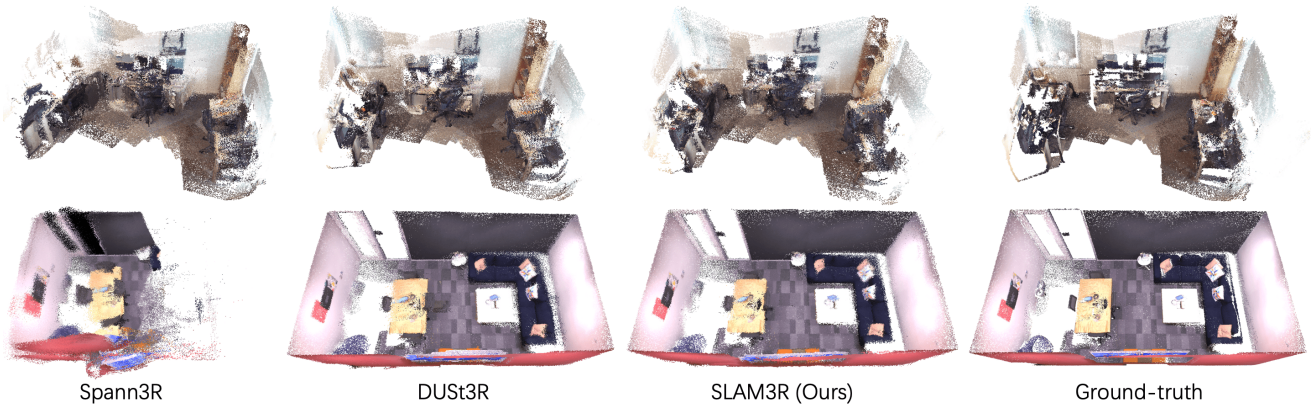

Figure 4. We visualize the reconstruction results on two scenes: Office-09 and Office 2 from the 7-Scenes [51] and Replica [54] datasets. Our method runs in real-time and achieves high-quality reconstruction comparable to the offline method DUSt3R [64].

4. Experiments

Datasets. For both the Image-to-Points (I2P) and Local-to-World (L2W) models, we perform training with a mixture of three datasets: ScanNet++ [71], Aria Synthetic Environments [3] and CO3D-v2 [41]. These datasets vary from scene-level to object-centric, and contain both real-world and synthetic scenes. Since they are all recorded sequentially, we can easily extract video clips with a sliding-window mechanism as our training data. We select about 850K clips for training in total. To validate our reconstruction quality, we conduct quantitative evaluations on two unseen datasets: 7 Scenes [51], a real-world dataset of partial scenes, and Replica [54], a synthetic dataset of complete scenes. We also demonstrate visual reconstruction results across diverse datasets and in-the-wild captured videos to showcase the generalization ability of SLAM3R.

Implementation details. Both the I2P and L2W models build upon the architecture of DUSt3R [64] with minimal but effective modifications, making it natural for them to

Figure 5. Qualitative examples. We show our reconstruction results on Tanks and Temples [26], BlendedMVS [69], Map-free Reloc [2], LLFF [35], and ETH3D [49, 50] datasets, as well as in-the-wild captured videos, to demonstrate SLAM3R's generalization ability.

initialize their weights from the DUSt3R pre-trained model. We initialize our weights using the DUSt3R model trained on 224×224 resolution images with $m = 24$ encoder blocks, $n = 12$ decoder blocks with linear heads. All images are center-cropped to 224×224 pixels before feeding into SLAM3R. Our training is conducted on 8 NVIDIA 4090D GPUs, each with 24 GB of memory. It takes about one day. At test time, we set the initial window length to $L = 5$ to ensure high-quality reconstruction of all frames within the window. For subsequent incremental windows, we use $L = 11$ to provide more supporting views for better keyframe reconstruction. Please refer to our supplementary material for more implementation details.

4.1. Comparisons

Evaluation metrics. Following NICER-SLAM [79] and Spann3R [61], we build a ground-truth point cloud model for each test sequence by back-projecting pixels to the world using ground-truth depths and camera parameters. The reconstructed point clouds are aligned to ground truths using Umeyama [58] and ICP [43] algorithms. We quantify reconstruction quality through accuracy and completeness metrics. To demonstrate computational efficiency, we report frames per second (FPS) on a single NVIDIA 4090D GPU. We also evaluate camera poses using absolute trajectory error (ATE-RMSE). For detailed formulations of the metrics, please refer to the supplementary material.

Reconstruction results on the 7 Scenes [51] dataset. The numerical results of scene reconstruction quality are reported in Table 1. Following Spann3R [61]'s setting, we uniformly sample one-twentieth of the frames in each test sequence as input video. Each video is regarded as an individual scene. We evaluate SLAM3R using two settings: integrating the full pointmaps predicted for all input frames to create reconstruction results (denoted by SLAM3R-NoConf), and filtering pointmaps with a confidence threshold of 3 before creating reconstruction results (SLAM3R). We compare our method with optimization-based reconstruction DUSt3R [64], triangulation-based MASt3R [28], and online incremental reconstruction Spann3R. Our method outperforms all baselines in both accuracy and completeness while maintaining real-time performance. As shown in the Office-09 scene (the top row in Figure 4), our approach demonstrates much less drift compared to the concurrent work Spann3R [61].

Reconstruction results on the Replica [54] dataset. Besides the baselines mentioned in 7 Scene datasets, we also compare the SLAM-based reconstruction approaches NICER-SLAM [79], DROID-SLAM [56], DIM-SLAM [29] and GO-SLAM [75] on the Replica [54] dataset. The numerical results on full scene reconstruction are reported in Table 2. Due to the memory constraint, DUSt3R [64] and MASt3R [28] process only one-twentieth of the frames for reconstruction. As is shown in the table, our method surpasses all baselines with FPS greater than 1. Notably, without any optimization procedure, our method achieves reconstruction quality comparable to optimization-based methods such as NICER-

Method	# Frames	Acc.	Comp.	FPS
DUSt3R [64]	2	3.16	2.89	42.55
I2P	2	3.39	3.04	42.55
I2P	5	2.62	2.28	40.82
I2P (Default)	11	2.38	2.03	40.11
I2P	15	2.27	1.94	35.51
I2P	51	**2.23**	**1.86**	11.97

Table 4. Inner-window keyframe reconstruction results with various window lengths. By default, we use 11-frame windows for incremental reconstruction to balance quality and efficiency.

SLAM [79] and DUSt3R [64]. Example of the Office 2 (the bottom row in Figure 4) also illustrates the global consistency of our reconstruction result.

Camera pose estimation on 7 Scenes [51] and Replica [54]. Our method is designed in a new paradigm that reconstructs 3D points end-to-end without explicitly solving camera parameters. Following DUSt3R [64], We also derive camera poses from the predicted scene points using PnP-RANSAC solver in OpenCV [6] with ground truth camera intrinsics of each frame. The results are reported in Table 3. We can observe that camera poses and scene reconstruction results are not fully positively correlated. This discrepancy between pose and reconstruction errors indicates that effective end-to-end 3D reconstruction is possible and promising without first obtaining precise camera poses.

For more details on the comparisons in this section, please refer to our supplementary material.

4.2. Analyses

Effectiveness of the I2P model. To highlight the advantages of our multi-view I2P model over the original two-view DUSt3R [64], we evaluate the reconstruction quality of keyframes with varying numbers of supporting views. We conduct experiments on the Replica [54] dataset, where input views are sampled using a sliding window of different sizes, and the reconstruction accuracy and completeness of the keyframes are computed. The results are reported in Table 4. As the number of supporting views increases, our approach progressively improves reconstruction quality. Notably, the efficiency of our method remains stable until the window size exceeds 11, demonstrating the effectiveness of our parallel design. However, the results also show diminishing returns as the number of views increases, which we detail in the supplementary material. Visual results of I2P reconstruction can be found in Figure 1.

Advantages of the L2W model. The effectiveness of the L2W model is evaluated through ablation studies on the Replica [54] dataset. Per-window reconstructions are first generated with a window size of 11 using the I2P model. Local points are then aligned to a unified coordinate frame using different methods: global optimization from DUSt3R [64] (I2P-GA), traditional approaches such as Umeyama [58] and ICP [43] (I2P+UI), and our L2W model (I2P+L2W+Re). For consistency, we set the window size for global optimization to 10, which is equal to the number of views used to align new frames in other methods. Results in Table 5 show that our full method achieves superior alignment accuracy and computational efficiency compared to the alternatives.

Method	Acc.	Comp.	FPS
I2P+GA	4.87	3.00	∼3
I2P+UI	7.47	3.86	∼1
I2P+L2W	6.19	3.54	∼92
I2P+L2W+Re (Full)	**3.62**	**2.70**	∼43

Table 5. Reconstruction results using various point alignment methods and scene frame selection strategies. The FPS reported only accounts for the overhead of the alignment operation.

Analysis of the retrieval module. We propose a lightweight retrieval module that selects historical scene frames from the reservoir. This approach effectively performs implicit re-localization. We compare our retrieval method with a baseline approach that selects the ten nearest previous frames, named I2P+L2W. The results in Table 5 indicate a significant performance improvement with our retrieval strategy, demonstrating its effectiveness.

In-the-wild scene reconstruction. We have tested our method on a diverse range of unseen datasets and found that SLAM3R shows strong generalization capabilities. Figure 5 shows our reconstruction results on Tanks and Temples [26], BlendedMVS [69], Map-free Reloc [2], LLFF [35], and ETH3D [49, 50] datasets, as well as in-the-wild videos we captured. These results show that our method performs reliably across different scales and scenes. We also provide additional numerical results on sampled scenes from these datasets in the supplementary material.

5. Conclusion

In this paper, we present SLAM3R, a novel and effective system that performs real-time, high-quality, dense 3D scene reconstruction using RGB videos. It employs a two-hierarchy neural network framework to perform end-to-end 3D reconstruction through streamlined feed-forward processes, eliminating the need to explicitly solve any camera parameters. Experiments demonstrate its state-of-the-art reconstruction quality and real-time efficiency (20+ FPS).

Limitations and future work. The elimination of camera parameter prediction prevents us from performing global bundle adjustment. Additionally, the poses derived from our scene point cloud prediction still fall short of SLAM systems that specialize in camera localization. Addressing these limitations will be the focus of our future work.

6. Acknowledgments

This work is supported by the National Key R&D Program of China (2022ZD0160800). This work is also supported by the Early Career Scheme of the Research Grants Council (grant # 27207224), the HKU-100 Award, a donation from the Musketeers Foundation, and in part by the JC STEM Lab of Robotics for Soft Materials funded by The Hong Kong Jockey Club Charities Trust. We thank the reviewers for their valuable feedback, as well as Zihan Zhu and Songyou Peng for their help with our experiments. Siyan Dong would also like to thank the support from HKU School of Computing and Data Science.

Supplementary Material

In this appendix, we first present additional implementation details and experimental settings in Sec. A and Sec. B, which were omitted from the main paper due to page limit. We then report additional analyses in Sec. C. Finally, we show more reconstruction results of our method in Sec. D.

A. Implementation details

Retrieval module. We propose a lightweight module for efficient scene frame retrieval to support the keyframe registration. The retrieval module directly reuses I2P's decoder blocks as its backbone, followed by a linear projection and an average-pooling layer. Specifically, it uses the first two blocks from both the supporting and keyframe decoders, for scene frames and keyframes (awaiting registration), respectively. It takes as input image features of one keyframe and all the scene frames in the buffering set, predicting correlation scores between the keyframe and each buffering frame. Notably, the correlation scores share similar behavior with the mean confidence of the I2P model's final prediction and offer unique advantages over the cosine similarity between image features of two frames. These correlation scores account for both visual similarity and provide suitable baselines for 3D reconstruction.

The module inherits the weights of the first two layers of the decoder in I2P model. During training, only the weights of the linear projection are updated using an L1 loss:

$$\mathcal{L}_{Retr} = \sum_{i=1}^{R} |S'_i - Mean(C'_i)|,$$
$$S'_i = Sigmoid(S_i),$$
$$C'_i = (C_i - 1)/C_i,$$

where R is the number of input supporting frames, S_i is the predicted correlation score between supporting frame i and the keyframe, C_i is the predicted confidence from the complete I2P model. Both S_i and C_i are normalized to [0,1] before calculating the loss.

Multi-keyframe co-registration. In practice, our scene decoder in the L2W model adopts the same architecture as the keyframe decoder in the I2P model, allowing for the simultaneous input and registration of multiple keyframes. In the decoding stage, scene frames and keyframes exchange information bidirectionally: each scene frame queries features from all keyframes, and each keyframe interacts with all scene frames. Compared to single-keyframe registration, this extension significantly reduces computational overhead by registering multiple keyframes with a single pass of the scene decoder. Furthermore, incorporating information from additional keyframes enhances the refinement of scene frame features, leading to more accurate reconstruction for all input frames.

Training details. To construct the training data, we utilize all iPhone and DSLR frames registered by COLMAP [47] from the training splits of ScanNet++[71]. Additionally, we include all frames from the first 450 scenes of the Aria Synthetic Environments (ASE)[3] dataset and 41 categories from CO3D-v2 [41], with each category containing up to 50 randomly sampled scene sequences. We introduce two ways to extract video clips for training. For ScanNet++ and ASE, we adopt uniform sampling with strides of 3 and 2, respectively. For CO3D-v2, frames are randomly sampled within temporal segments covering half the length of each video. In total, we extract approximately 850K clips. During each epoch of training, we randomly sample 4000, 2000, and 2000 clips from the ScanNet++, ASE, and CO3D-v2 datasets, respectively. All training images are resized and then center-cropped to 224 × 224 pixels. Standard data augmentation techniques [64] are applied.

To train our I2P model, we extend the training process of DUSt3R from two views to multiple views. Specifically, our I2P model takes as input a video clip of length 11, and designates the middle frame as the keyframe. We train the I2P model for 100 epochs, which takes about 6 hours. After that, we train the retrieval module built on the I2P model. During training, we freeze all other modules and use L1 loss to supervise the correlation score against the mean confidence of the I2P model's final predictions. This module requires 50 epochs of training, which takes about 2 hours.

To train the L2W model, we use clips of length 12, with the first six images selected as scene frames, and the last six images designated as keyframes to register. The model is trained for 200 epochs in total, and the training process takes approximately 16 hours. When training with ground truth pointmaps as input, we set invalid points to (0,0,0). A confidence-aware loss without scale normalization is applied, ensuring that the predicted point maps retain consistent scale with the input scene frames.

Our training is conducted on 8 NVIDIA 4090D GPUs, each with 24GB of memory and a batch size of 4 per GPU.

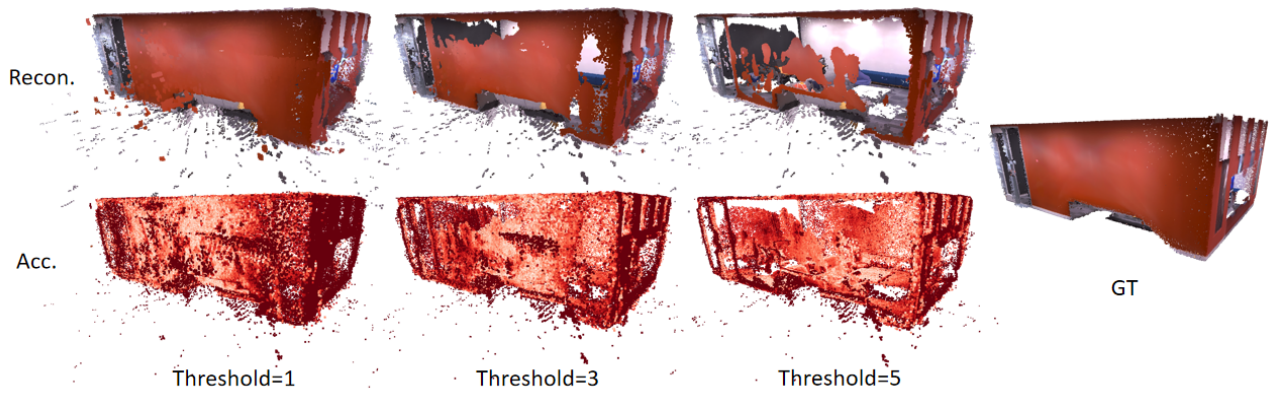

Figure 6. The reconstruction results and the corresponding accuracy heatmaps of MASt3R [28] on Office 3 from Replica [54] dataset under different confidence thresholds. Lighter colors indicate higher accuracy.

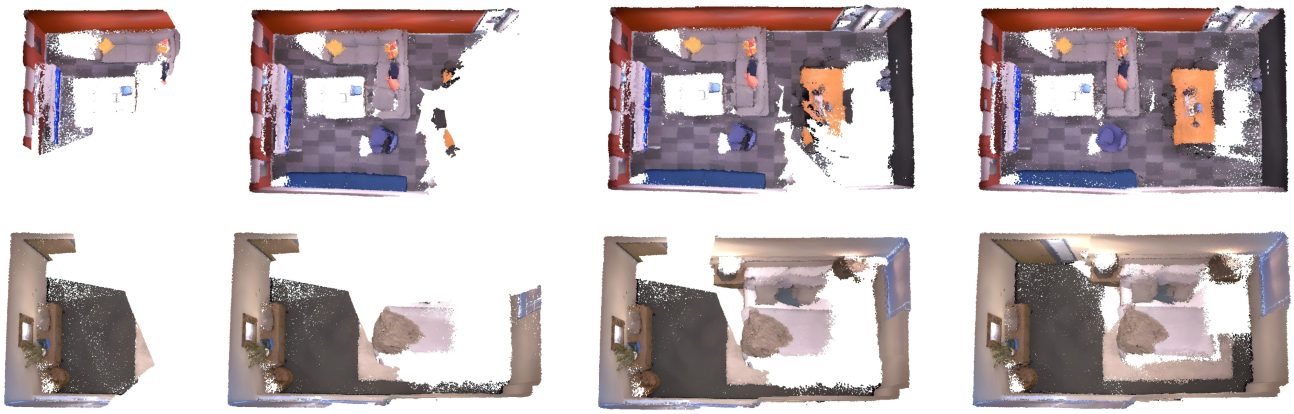

Figure 7. Visualization of the incremental reconstruction process of our method on the Office 3 and Room 1 of Replica [54] dataset. Our method achieves low drift without any global-optimization stage.

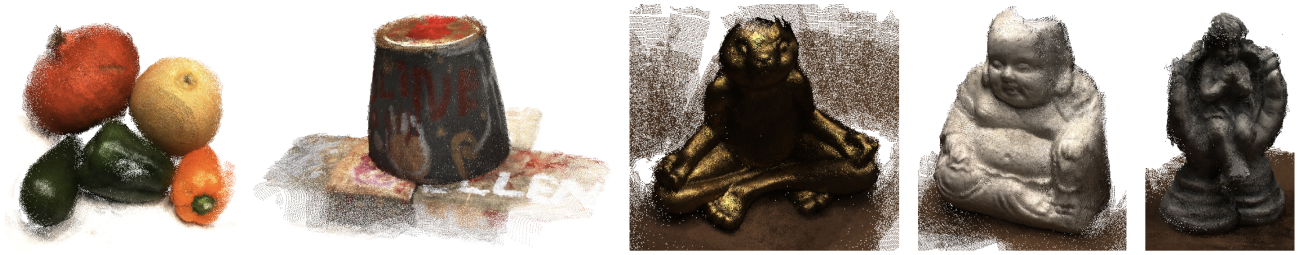

Figure 8. Reconstruction results on unorganized image collections from DTU [1] dataset.

B. Details for experimental settings

Calculation of the evaluation metrics. To evaluate reconstruction quality, we use accuracy and completeness as our metrics. They are calculated by:

$$Accuracy = \frac{1}{P}\sum_{i=1}^{P} min_j(D(x_i, y_j)),$$

$$Completeness = \frac{1}{Q}\sum_{j=1}^{Q} min_i(D(x_i, y_j)).$$

P and Q are the numbers of points in the reconstructed point cloud and GT point cloud respectively. $D(\cdot)$ represents Euclidean distance, and x_i and y_j represent iterating each point from the reconstructed and GT point cloud.

To measure the efficiency, we report FPS (frames per

Method	Chess	Fire	Heads	Office	Pumpkin	RedKitchen	Stairs	Average	FPS
DUSt3R [64] (w/PnP)	5.09	4.88	2.52	12.07	**10.64**	10.35	10.55	8.02	<1
MASt3R [28] (w/PnP)	4.32	**2.92**	**1.47**	12.37	11.82	7.98	**3.04**	**6.28**	≪1
NICER-SLAM [79]*	**3.28**	6.85	4.16	**10.84**	20.00	**3.94**	10.81	8.55	<1
DROID-SLAM [56]*	**3.36**	**2.40**	**1.43**	**9.19**	16.46	**4.94**	**1.85**	**5.66**	∼20
Spann3R [61]	9.18	6.69	7.10	21.56	12.83	14.06	10.43	11.70	>50
SLAM3R-NoConf (Ours)	6.29	5.33	4.47	12.42	11.74	9.53	9.30	8.44	∼25
SLAM3R (Ours)	6.20	5.30	4.56	12.40	**11.71**	9.47	9.20	8.41	∼25

Table 6. Camera pose estimation results on the 7Scenes [51] dataset reported using the ATE-RMSE (cm) metric. The average numbers are computed over all test scenes. * denotes the results reported in NICER-SLAM.

Method	Room 0	Room 1	Room 2	Office 0	Office 1	Office 2	Office 3	Office 4	Average	FPS
DUSt3R [64] (w/PnP)	4.00	4.49	7.62	4.88	4.04	3.90	2.84	6.30	4.76	<1
MASt3R [28] (w/PnP)	**1.07**	**0.99**	**0.87**	**0.90**	4.90	**1.21**	1.77	**1.63**	**1.67**	≪1
NICER-SLAM [79]*	1.36	1.60	1.14	2.12	**3.23**	2.12	**1.42**	2.01	1.88	<1
GO-SLAM [75]	-	-	-	-	-	-	-	-	0.39	∼8
DIM-SLAM [29]	0.48	0.78	0.35	0.67	**0.37**	0.36	**0.33**	**0.36**	0.46	∼3
DROID-SLAM [56]*	**0.34**	**0.13**	**0.27**	**0.25**	0.42	**0.32**	0.52	0.40	**0.33**	∼20
Spann3R [61]	29.76	34.78	26.08	34.50	22.65	34.47	42.24	37.84	32.79	>50
SLAM3R-NoConf (Ours)	4.54	5.89	5.73	11.17	6.32	6.15	4.99	8.05	6.61	∼24
SLAM3R (Ours)	4.56	5.88	5.72	11.17	6.32	6.15	4.95	8.09	6.61	∼24

Table 7. Camera pose estimation results on the Replica [54] dataset reported using the ATE-RMSE (cm) metric.

second), which is calculated by:

$$FPS = F/time,$$

where $time$ is the total time used to reconstruct the scene, and F is the number of frames from the video.

We evaluate the camera pose accuracy using absolute trajectory error (ATE-RMSE), which is formulated by:

$$ATE\text{-}RMSE = \sqrt{\frac{1}{F}\sum_{i=1}^{F} D(T_i^{gt}, T_i^{perd})^2},$$

where T^{perd} and T^{gt} are the camera center positions of the predicted and GT camera trajectories.

Full video as input on Replica [54]. On the Replica dataset, we reconstruct the entire scene geometry using all video frames. With the stride of the sliding window set to 1, all frames will be used as a keyframe once. For each window, frames are sampled around the keyframe, with $Skip = 20$ frames per supporting frame, to ensure reasonable camera motion (disparity). We co-register $Co = 10$ keyframes at each time, which share the same $K = 10$ scene frames as a reference. These scene frames are selected through a two-step process. First, we calculate the correlation score between all frames in the buffering set and the Co keyframes. Then, we select K frames from the buffering set that show the highest total correlation score with these keyframes. After every $R = 20$ registered keyframes, we update the buffering set by retaining the keyframes with the highest reconstruction scores, where reconstruction score of a frame is the product of its mean confidence predicted by I2P and L2W model. The insertion/update follows the reservoir sampling probability described in the main paper.

Sampled frames as input on 7 Scenes [51]. Following Spann3R [61], the frames in each test sequence are sampled with a stride of 20, and we only reconstruct the points from the sampled frames. To handle sampled-frame-only input, we adapt our reconstruction pipeline for full-video input by setting $Skip = 1$, $Co = 2$, $K = 5$, and $R = 1$ in practice.

Experiments on DUSt3R [64] and MASt3R [28]. The global optimization with complete graph setting in DUSt3R and MASt3R requires substantial GPU memory. Consequently, to evaluate the global reconstruction quality of these two methods on the Replica dataset, we uniformly sample 1/20 of the images. DUSt3R is tested using the weight-224 model with a resolution of 224×224, the same as our input resolution, while MASt3R is tested using the weight-512 model with resolutions of 512×384 and 512×288 as inputs for reconstructing the 7 Scenes [51] and Replica [54] datasets, respectively. Note that a resolution of 224×224 results in less overlap between adjacent frames, making reconstruction inherently more challenging.

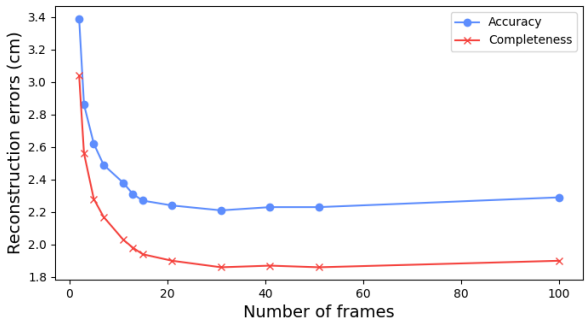

Figure 9. Inner-window keyframe reconstruction results from various window lengths.

During the evaluation, we observed that MASt3R occasionally generates floating points with high confidence scores, which are difficult to filter using confidence thresholds and significantly degrade accuracy. An example of this issue is shown in Figure 6. In contrast, our confidence scores are more effective and successfully reduce erroneous points. The results of SLAM3R reported on 7 Scenes and Replica datasets use a fixed confidence threshold of 3.

C. Additional comparisons and analyses

More numerical results. We report more quantitative comparisons of reconstruction results on ScanNet [11], Tanks and Temples [26], and ETH3D [50] datasets. We sampled three scenes from each dataset, and report the results in Table 8. SLAM3R outperforms Spann3R in most cases and demonstrates performance either comparable to or better than DUSt3R. These results further verify our method's effectiveness.

ScanNet	scene0011_00	scene0015_00	scene0019_00	Average
DUSt3R [64]	**5.56 / 3.76**	**5.04 / 4.10**	4.52 / 4.74	**5.04 / 4.20**
Spann3R [61]	13.09 / 11.37	8.51 / 7.79	7.97 / 9.66	9.86 / 9.61
SLAM3R (Ours)	5.86 / 3.98	5.98 / 5.97	**4.27 / 4.34**	5.37 / 4.76
Tanks and Temples	Ignitius	Truck	Caterpillar	Average
DUSt3R [64]	3.55/ 1.22	9.31 / **4.85**	12.67 / 5.25	8.51 / **3.77**
Spann3R [61]	5.51 / 1.10	6.40 / 12.61	**11.50**/ 5.74	7.80 / 6.48
SLAM3R (Ours)	**3.30 / 0.94**	**5.35**/ 5.59	12.26 / **5.05**	**6.97** / 3.86
ETH3D	plant_scene_1	table_3	sofa_1	Average
DUSt3R [64]	2.98 / 2.48	3.13 / **1.30**	**2.05** / 3.67	2.72 / 2.48
Spann3R [61]	2.54 / 4.25	3.03 / 2.08	2.10 / 4.55	2.56 / 3.62
SLAM3R (Ours)	**2.36 / 1.98**	**2.75** / 1.34	2.13 / **1.90**	**2.41 / 1.74**

Table 8. Reconstruction errors (accuracy / completeness) on ScanNet [11], Tanks and Temples [26], and ETH3D [50] datasets.

Diminishing return of window length. In the main paper, we report the I2P reconstruction results with different window lengths. Here, we further analyze the diminishing returns, which indicate that the window length should not be too large. As Figure 9 shows, the accuracy and completeness of the keyframe reconstruction improve rapidly at first as input frames increase, but then gradually decline. This is because larger windows result in less and less overlapping. Additionally, the inference time becomes significantly slower as length increases. Consequently, we set the window size to 11 in our main experiments, balancing the reconstruction quality and runtime efficiency.

# Scene frames	Acc.	Comp.	FPS
1	4.18	2.61	~398
5	3.99	2.79	~247
10	**3.57**	2.62	~152
20	**3.57**	2.60	~86
30	3.59	**2.58**	~61
40	4.15	3.05	~46
50	4.27	3.15	~37

Table 9. Reconstruction results on Replica [54] dataset, with various maximum number of scene frames selected for keyframe registration. The FPS of the L2W model aligning 10 keyframes at once with different numbers of input scene frames is also reported.

Effect of scene frame numbers on registration. We conduct experiments on the Replica [54] dataset to investigate how the number of scene frames selected as a global reference affects the registration quality of keyframes. As reported in Table 9, the accuracy of full-scene registration initially improves as the maximum number of input scene frames increases but eventually declines beyond a certain threshold. Retrieving too few scene frames from the buffering set risks missing suitable frames and causing keyframe registration to get stuck in local minimums. Conversely, selecting too many scene frames can introduce irrelevant ones that add noise and hinder registration.

To balance reconstruction accuracy and runtime efficiency, we set the number of retrieved scene frames to 5 and 10 on 7 Scenes [51] and Replica [54] dataset, which achieves consistent and reliable performance.

Camera pose estimation. The detailed results are presented in Table 6 and Table 7. For DUSt3R [64] and MASt3R [28], we evaluate the camera poses derived via the PnP-RANSAC solver with their predicted pointmaps (after global alignment) and GT intrinsic parameters. When evaluating Spann3R [61] on the Replica [54] dataset, only one-twentieth of the frames are used, as it fails to give reasonable results with all frames input.

We outperform the concurrent work Spann3R [61], demonstrating the effectiveness of our hierarchical design with multi-view input and global retrieval. Among classical SLAM systems, the pose errors of GO-SLAM [75] and DROID-SLAM [56] are lower than those of NICER-

SLAM. However, their reconstruction accuracy and completeness are worse. This discrepancy between pose and reconstruction errors indicates that effective end-to-end 3D reconstruction is possible and promising without first obtaining precise camera poses.

D. More visual results

Visualization of incremental reconstruction. Figure 7 visualizes the process of our incremental reconstruction on two scenes from Replica [54]. Our method achieves effective alignment at loops while experiencing minimal cumulative drift, without offline global optimization step.

Reconstruction on DTU [1] dataset. The results are shown in Figure 8. Note that our method does not require any camera parameters, and produces dense point cloud reconstructions end-to-end in real-time.

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120:153–168, 2016. 2, 10, 13

[2] Eduardo Arnold, Jamie Wynn, Sara Vicente, Guillermo Garcia-Hernando, Aron Monszpart, Victor Prisacariu, Daniyar Turmukhambetov, and Eric Brachmann. Map-free visual relocalization: Metric pose relative to a single image. In *European Conference on Computer Vision*, pages 690–708. Springer, 2022. 7, 8

[3] Armen Avetisyan, Christopher Xie, Henry Howard-Jenkins, Tsun-Yi Yang, Samir Aroudj, Suvam Patra, Fuyang Zhang, Duncan Frost, Luke Holland, Campbell Orme, et al. Scenescript: Reconstructing scenes with an autoregressive structured language model. *arXiv preprint arXiv:2403.13064*, 2024. 6, 9

[4] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE robotics & automation magazine*, 13(3):108–117, 2006. 2

[5] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568, 2018. 1, 2

[6] G Bradski. The opencv library. *Dr. Dobb's Journal of Software Tools*, 2000. 8

[7] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. 1, 2

[8] Yiyang Chen, Siyan Dong, Xulong Wang, Lulu Cai, Youyi Zheng, and Yanchao Yang. Sg-nerf: Neural surface reconstruction with scene graph optimization. *arXiv preprint arXiv:2407.12667*, 2024. 2

[9] Chi-Ming Chung, Yang-Che Tseng, Ya-Ching Hsu, Xiang-Qian Shi, Yun-Hung Hua, Jia-Fong Yeh, Wen-Chin Chen, Yi-Ting Chen, and Winston H Hsu. Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9400–9406. IEEE, 2023. 1, 2

[10] Jan Czarnowski, Tristan Laidlow, Ronald Clark, and Andrew J Davison. Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters*, 5 (2):721–728, 2020. 1

[11] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 2, 12

[12] Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2

[13] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[14] Bardienus Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. Mast3r-sfm: a fully-integrated solution for unconstrained structure-from-motion. *arXiv preprint arXiv:2409.19152*, 2024. 3

[15] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006. 2

[16] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014. 1, 2

[17] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. 1, 2

[18] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015. 1, 2

[19] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024. 2

[20] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2

[21] Huajian Huang, Longwei Li, Hui Cheng, and Sai-Kit Yeung. Photo-slam: Real-time simultaneous localization and photo-realistic mapping for monocular stereo and rgb-d cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21584–21593, 2024. 2

[22] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu. Di-fusion: Online implicit 3d reconstruction with

deep priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8932–8941, 2021. 1, 2

[23] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21357–21366, 2024. 2

[24] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1

[25] Georg Klein and David Murray. Parallel tracking and mapping on a camera phone. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 83–86. IEEE, 2009. 1, 2

[26] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 7, 8, 12

[27] Lukas Koestler, Nan Yang, Niclas Zeller, and Daniel Cremers. Tandem: Tracking and dense mapping in real-time using deep multi-view stereo. In *Conference on Robot Learning*, pages 34–45. PMLR, 2022. 2

[28] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024. 2, 3, 6, 7, 10, 11, 12

[29] Heng Li, Xiaodong Gu, Weihao Yuan, Luwei Yang, Zilong Dong, and Ping Tan. Dense rgb slam with neural implicit maps. In *Proceedings of the International Conference on Learning Representations*, 2023. 1, 2, 6, 7, 11

[30] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023. 2

[31] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5987–5997, 2021. 1, 2

[32] Lorenzo Liso, Erik Sandström, Vladimir Yugay, Luc Van Gool, and Martin R Oswald. Loopy-slam: Dense neural slam with loop closures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20363–20373, 2024. 2

[33] Shaohui Liu, Yifan Yu, Rémi Pautrat, Marc Pollefeys, and Viktor Larsson. 3d line mapping revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21445–21455, 2023. 1, 2

[34] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18039–18048, 2024. 2

[35] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (ToG)*, 38(4):1–14, 2019. 7, 8

[36] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[37] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 1, 2

[38] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1, 2

[39] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011. 1, 2

[40] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 523–540. Springer, 2020. 1

[41] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10901–10911, 2021. 6, 9

[42] Antoni Rosinol, John J Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3437–3444. IEEE, 2023. 1, 2

[43] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE, 2001. 7, 8

[44] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18433–18444, 2023. 2

[45] Erik Sandström, Kevin Ta, Luc Van Gool, and Martin R Oswald. Uncle-slam: Uncertainty learning for dense neural slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4537–4548, 2023. 2

[46] Erik Sandström, Keisuke Tateno, Michael Oechsle, Michael Niemeyer, Luc Van Gool, Martin R Oswald, and Federico Tombari. Splat-slam: Globally optimized rgb-only slam with 3d gaussians. *arXiv preprint arXiv:2405.16544*, 2024. 1, 2

[47] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 1, 2, 9

[48] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for

unstructured multi-view stereo. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 501–518. Springer, 2016. 1, 2

[49] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3260–3269, 2017. 7, 8

[50] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144, 2019. 7, 8, 12

[51] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2937, 2013. 6, 7, 8, 11, 12

[52] Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibarated image pairs. *arXiv preprint arXiv:2408.13912*, 2024. 2

[53] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006. 1, 2

[54] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 6, 7, 8, 10, 11, 12, 13

[55] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6229–6238, 2021. 1, 2

[56] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 1, 2, 6, 7, 11, 12

[57] Fabio Tosi, Youmin Zhang, Ziren Gong, Erik Sandström, Stefano Mattoccia, Martin R Oswald, and Matteo Poggi. How nerfs and 3d gaussian splatting are reshaping slam: a survey. *arXiv preprint arXiv:2402.13255*, 4, 2024. 1

[58] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380, 1991. 7, 8

[59] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985. 5

[60] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, and Marc Pollefeys. Itermvs: Iterative probability estimation for efficient multi-view stereo. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8606–8615, 2022. 1, 2

[61] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 2, 3, 6, 7, 11, 12

[62] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 1, 2

[63] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. *arXiv preprint arXiv:2410.19115*, 2024. 2

[64] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024. 1, 2, 3, 4, 6, 7, 8, 9, 11, 12

[65] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3295–3306, 2023. 2

[66] Changchang Wu. Visualsfm: A visual structure from motion system. *http://www. cs. washington. edu/homes/ccwu/vsfm*, 2011. 1, 2

[67] Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19595–19604, 2024. 2

[68] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 1, 2

[69] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1790–1799, 2020. 7, 8

[70] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. *arXiv preprint arXiv:2410.24207*, 2024. 2

[71] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 6, 9

[72] Vladimir Yugay, Yue Li, Theo Gevers, and Martin R Oswald. Gaussian-slam: Photo-realistic dense slam with gaussian splatting. *arXiv preprint arXiv:2312.10070*, 2023. 2

[73] Ganlin Zhang, Erik Sandström, Youmin Zhang, Manthan Patel, Luc Van Gool, and Martin R Oswald. Glorie-slam: Globally optimized rgb-only implicit encoding point cloud slam. *arXiv preprint arXiv:2403.19549*, 2024. 1, 2

[74] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024. 2

[75] Youmin Zhang, Fabio Tosi, Stefano Mattoccia, and Matteo Poggi. Go-slam: Global optimization for consistent 3d instant reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3727–3737, 2023. 1, 2, 6, 7, 11, 12

[76] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. Deeptam: Deep tracking and mapping. In *Proceedings of the European conference on computer vision (ECCV)*, pages 822–838, 2018. 1, 2

[77] Heng Zhou, Zhetao Guo, Shuhong Liu, Lechen Zhang, Qihao Wang, Yuxiang Ren, and Mingrui Li. Mod-slam: Monocular dense mapping for unbounded 3d scene reconstruction. *arXiv preprint arXiv:2402.03762*, 2024. 1, 2

[78] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12786–12796, 2022. 1, 2

[79] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. In *2024 International Conference on 3D Vision (3DV)*, pages 42–52. IEEE, 2024. 1, 2, 6, 7, 8, 11